Abyan Ardiatama

24060120140161

Praktikum 3-4 ML

Mengimport library yang dibutuhkan

```
%matplotlib inline
import matplotlib.pyplot as plt
plt.style.use('seaborn-whitegrid')
import numpy as np
from sklearn.cluster import KMeans
```

Mengimport dataset Iris

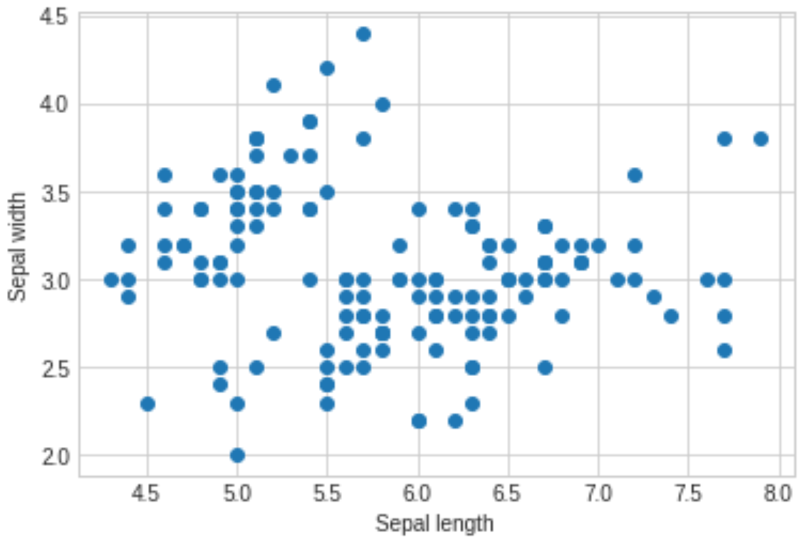
```
# Get iris dataset
from sklearn.datasets import load_iris
iris = load_iris()
X = iris.data
```

Clustering Feature 1 dan 2

```
# Get feature 1 and 2
X1 = X[:,0]
X2 = X[:,1]
X12 = np.array(list(zip(X1, X2)))
```

##

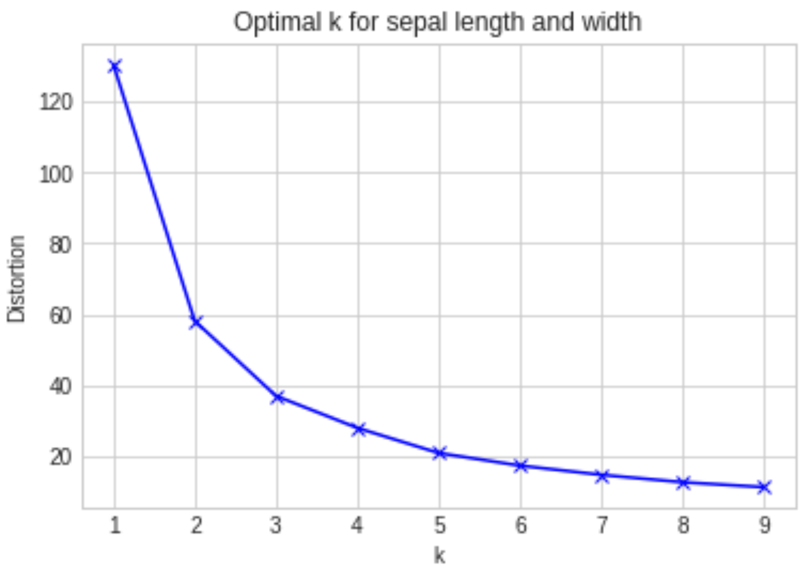
```
# Plot the data
plt.scatter(X1, X2)
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')
plt.show()
```



Tentukan nilai K dengan elbow method

```
# Elbow method
distortions = []
K = range(1,10)
for k in K:
    kmeanModel = KMeans(n_clusters=k)
    kmeanModel.fit(X12)
    distortions.append(kmeanModel.inertia_)
```

```
# Plot the elbow
plt.plot(K, distortions, 'bx-')
plt.xlabel('k')
plt.ylabel('Distortion')
plt.title('Optimal k for sepal length and width')
plt.show()
```



Cluster K-Means dengan K=3

```
# Cluster the data
kmeans = KMeans(n_clusters=3)
kmeans.fit(X12)
y_kmeans = kmeans.predict(X12)
```

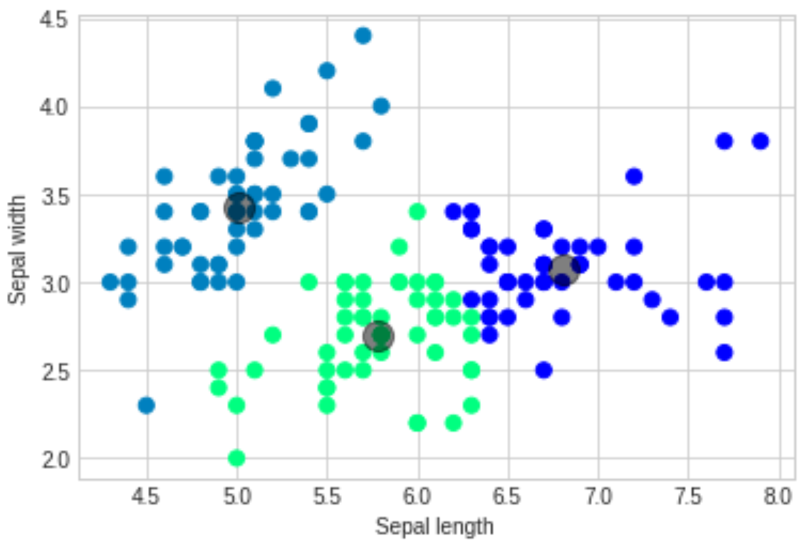
Evaluasi

```
# Evaluate the clustering
from sklearn import metrics
print("Homogeneity: %0.3f" % metrics.homogeneity_score(iris.target, y_kmeans))
print("Completeness: %0.3f" % metrics.completeness_score(iris.target, y_kmeans))
print("V-measure: %0.3f" % metrics.v_measure_score(iris.target, y_kmeans))
print("Adjusted Rand Index: %0.3f"
        % metrics.adjusted_rand_score(iris.target, y_kmeans))
print("Adjusted Mutual Information: %0.3f"
        % metrics.adjusted_mutual_info_score(iris.target, y_kmeans))
print("Silhouette Coefficient: %0.3f"
        % metrics.silhouette_score(X12, y_kmeans))
```

```
Homogeneity: 0.646
Completeness: 0.647
V-measure: 0.647
Adjusted Rand Index: 0.601
Adjusted Mutual Information: 0.642
Silhouette Coefficient: 0.445
```

Visualisasi hasil clustering feature 1 & 2

```
# Plot the clusters
plt.scatter(X1, X2, c=y_kmeans, s=50, cmap='winter')
centers = kmeans.cluster_centers_
plt.scatter(centers[:, 0], centers[:, 1], c='black', s=200, alpha=0.5)
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')
plt.show()
```



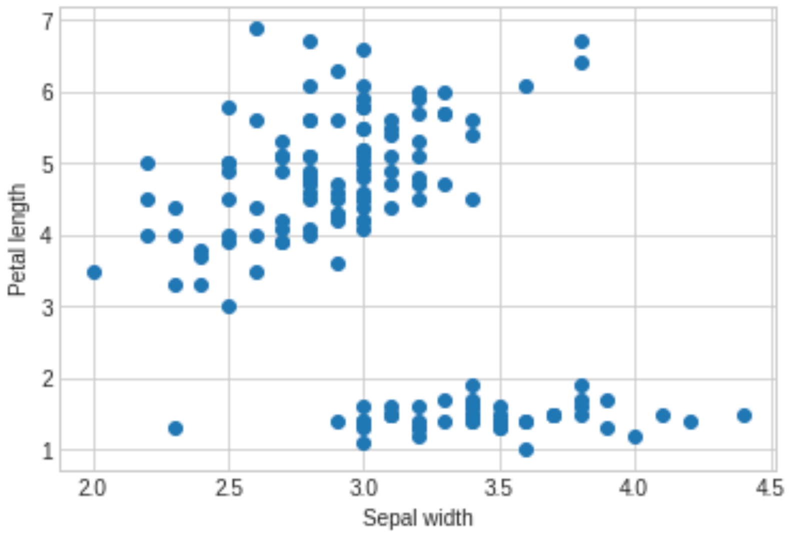
Clustering Feature 2 dan 3

```
# Feature 2 and 3
X2 = X[:,1]
X3 = X[:,2]

X23 = np.array(list(zip(X2, X3)))
```

Double-click (or enter) to edit

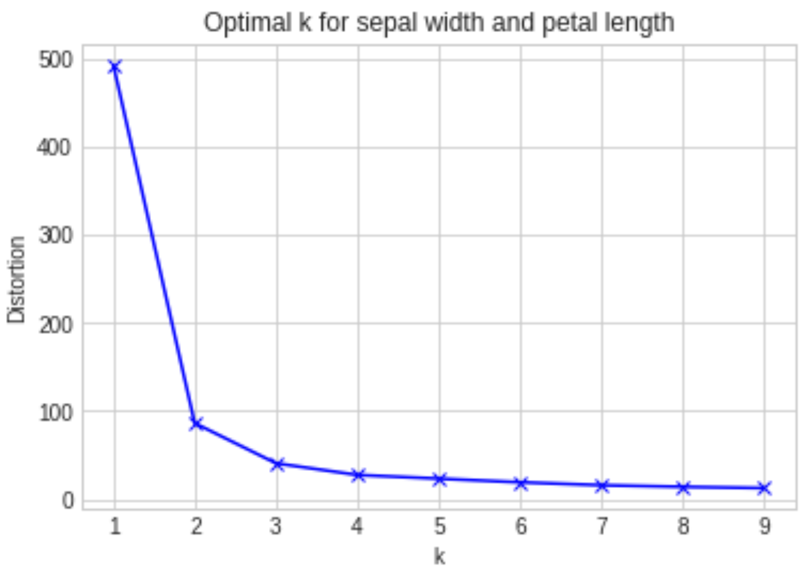
```
# Plot the data
plt.scatter(X2, X3)
plt.xlabel('Sepal width')
plt.ylabel('Petal length')
plt.show()
```



Menentukan nilai K dengan Elbow Method

```
# Elbow method
distortions = []
K = range(1,10)
for k in K:
    kmeanModel = KMeans(n_clusters=k)
    kmeanModel.fit(X23)
    distortions.append(kmeanModel.inertia_)


# Plot the elbow
plt.plot(K, distortions, 'bx-')
plt.xlabel('k')
plt.ylabel('Distortion')
plt.title('Optimal k for sepal width and petal length')
plt.show()
```



Clustering dengan K=2

```
# Cluster the data
kmeans = KMeans(n_clusters=2)
kmeans.fit(X23)
y_kmeans = kmeans.predict(X23)
```

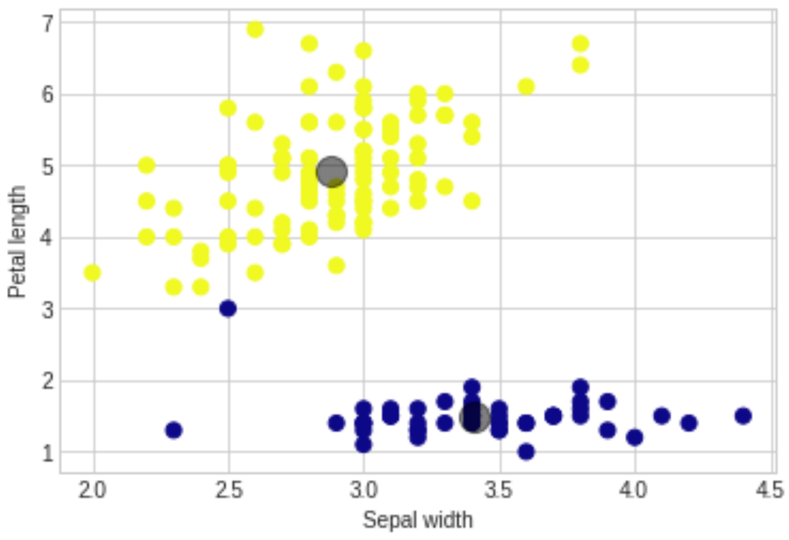
Evaluasi

```
# Evaluate the clustering
from sklearn import metrics
print("Homogeneity: %0.3f" % metrics.homogeneity_score(iris.target, y_kmeans))
print("Completeness: %0.3f" % metrics.completeness_score(iris.target, y_kmeans))
print("V-measure: %0.3f" % metrics.v_measure_score(iris.target, y_kmeans))
print("Adjusted Rand Index: %0.3f"
        % metrics.adjusted_rand_score(iris.target, y_kmeans))
print("Adjusted Mutual Information: %0.3f"
        % metrics.adjusted_mutual_info_score(iris.target, y_kmeans))
print("Silhouette Coefficient: %0.3f"
        % metrics.silhouette_score(X23, y_kmeans))
```

```
Homogeneity: 0.554
Completeness: 0.949
V-measure: 0.699
Adjusted Rand Index: 0.558
Adjusted Mutual Information: 0.697
Silhouette Coefficient: 0.739
```

Visualisasi

```
# Plot the clusters
plt.scatter(X2, X3, c=y_kmeans, s=50, cmap='plasma')
centers = kmeans.cluster_centers_
plt.scatter(centers[:, 0], centers[:, 1], c='black', s=200, alpha=0.5)
plt.xlabel('Sepal width')
plt.ylabel('Petal length')
plt.show()
```

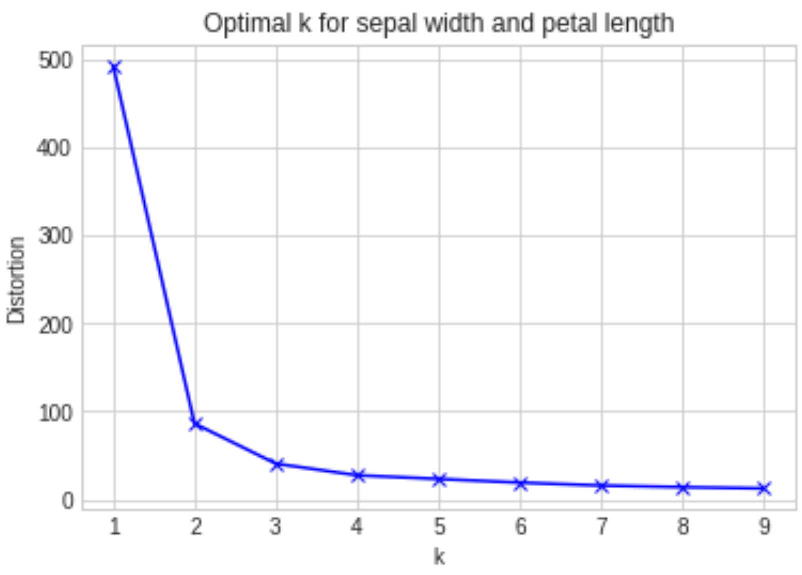


Perbedaan bx-plot dan x- plot

Tampilan bx- plot

b --> warna x --> tanda x di tiap titik '-' --> garis sepanjang grafik

```
# Tampilan bx- plot
plt.plot(K, distortions, 'bx-')
plt.xlabel('k')
plt.ylabel('Distortion')
plt.title('Optimal k for sepal width and petal length')
plt.show()
```

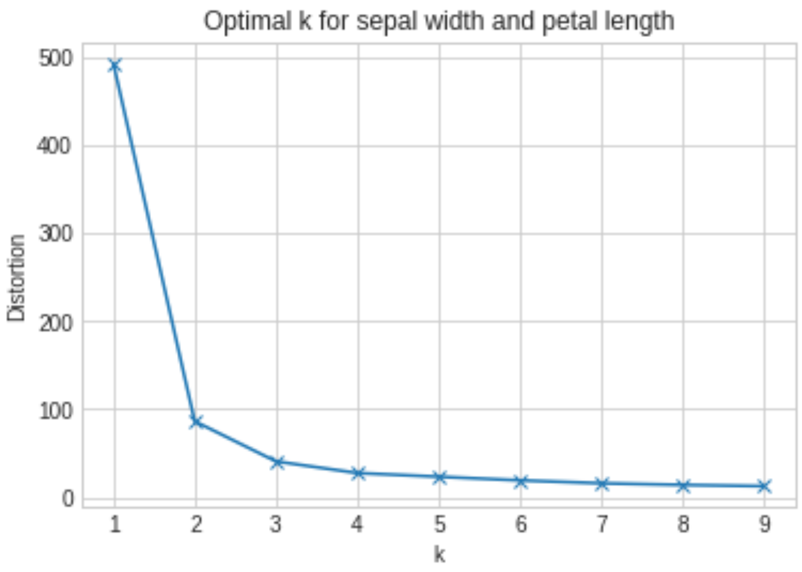


Tampilan x-plot

x --> tanda x di tiap titik

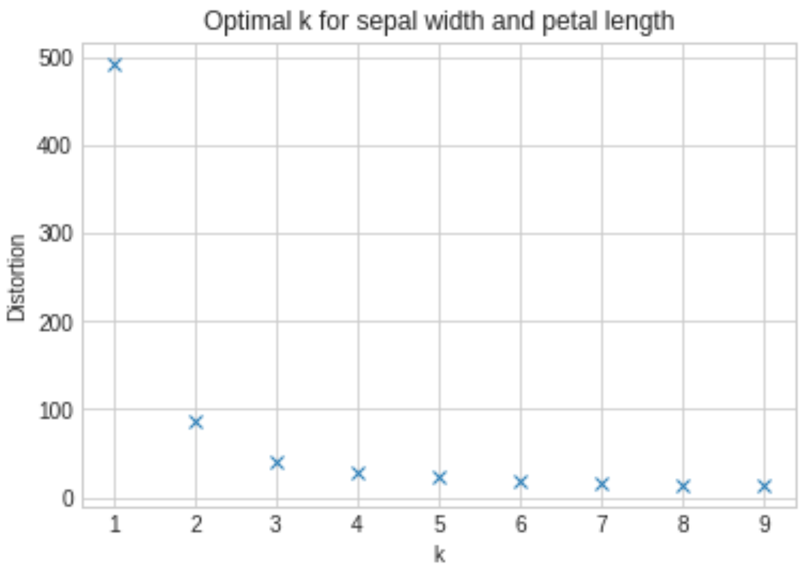
- --> garis di sepanjang grafik

```
# Tampilan x- plot
plt.plot(K, distortions, 'x-')
plt.xlabel('k')
plt.ylabel('Distortion')
plt.title('Optimal k for sepal width and petal length')
plt.show()
```



Tampilan x plot x --> tanda x di tiap titik

```
# Tampilan x plot
plt.plot(K, distortions, 'x')
plt.xlabel('k')
plt.ylabel('Distortion')
plt.title('Optimal k for sepal width and petal length')
plt.show()
```


Optimal k for sepal width and petal length
Distortion
k